An Object Oriented Approach for Data Fusion

Kaeye Dästner[1], Thomas Kausch[1], Felix Opitz[2]

[1]Systems for Surface Ships
ATLAS ELEKTRONIK GmbH
Wörthstr. 85
89077 Ulm, Germany
kaeye.daestner@atlas-elektronik.com
thomas.kausch@atlas-elektronik.com

[2]Defence and Communications Systems
EADS
Wörthstr. 85
89077 Ulm, Germany
felix.opitz@eads.com

Abstract: An new object oriented development suite for data fusion is presented. It is shown how the various issues in the data fusion development like design, implementation, simulation and testing may be supported. This allows the realisation of high sophisticated data fusion systems as applied in numerous civil and defence areas: e.g. air traffic control, coastal surveillance, vehicle guidance systems, or safety and defence applications. The approach supports the design of multi layer data fusion systems with different architecture integrating numerous different information sources. Besides of the generation of an object oriented data fusion kernel the developer is also supported in rapid prototyping just at the beginning of the developing process. Further the final software and system testing and verification within a simulated and real environment is taken into account.

1 Introduction

There are many contribution to the development of data fusion systems, which lead to a variety of concepts based on the experience with multiple projects [BS01], [OHK04], [OK05]. Today's multisensor data fusion systems have to integrate very diverse sensor suites. The information sources may be sensors like radars, electro-optical sensors, or acoustic sensors. Other sources of information have also to be integrated into the data fusion process: Tactical data links, sensor networks (e.g. air traffic management - ATC), or C² systems. Furthermore, according to the individual application the communication infrastructures vary from fibre optic cable to mobile radio network. The multisensor data fusion described in this paper deals with the tracking, classification and identification topic. Hence, the data fusion development suite must be able to generate a real-time data fusion system, which consists of all the algorithms suitable for the concrete application.

2 Object Oriented Data Fusion Kernel

First the data fusion kernel and its environment consisting of middleware and interfaces will be described bellow. A data fusion system is always integrated into a middleware specific to the application. This middleware provides a real-time database, which offers the input data to the data fusion and receives the data fusion results afterwards.

Of course there exists numerous realisations of such a middleware using different mechanisms. The interfaces are separated from the pure data fusion kernel through the declaration of an abstract interface class, which is responsible for the transformation of external data formats into the fusion kernel internal one (figure 1).

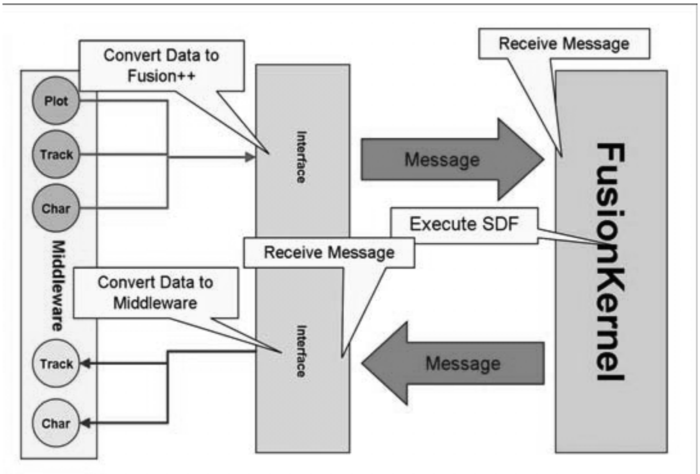

Figure 1: Middleware, interfaces and fusion kernel

A approach to design complex data fusion systems is to split the fusion process into multiple fusion nodes [BS01], [FHK04], [DKO07]. Therefore one constructs a (directed) fusion tree (figure 2), which condenses information more and more until the information suitable for an operator is delivered. The fusion is performed in the different nodes of this tree. Each node contains the functionalities data alignment, data association, and state estimation. The structure of this fusion tree depends on different interdependent operational and technical constraints.

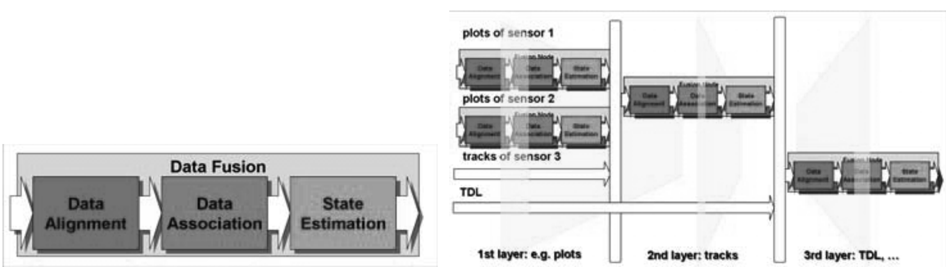

Figure 2: Multilayer data fusion: Fusion node and fusion tree.

2.1 Data alignment

Data alignment contains the topics of data storing, bias corrections and coordinate transformations. Therefore, all relevant coordinate systems and transformations are offered, i.e. Cartesian and polar systems with different origins (e.g. WGS84, …).

2.2 Data association

Data association examines the relations between sensor data and the objects specified by the application. A sensor data may be caused by one or multiple objects, or by the environment. On the other side, a object may also be not detected by the sensor for several reasons. The data association generates, evaluates and select hypothesis about the associations between sensor data and the phenomena which originates them. Examples of such algorithms are (figure 3): nearest neighbour, multidimensional data association (Lagrange Relaxation, LP) respectively MHT, or statistical methods (JPDA).

2.3 Estimation

The estimation part includes algorithms which estimate the object kinematics, class and identity. For the estimation of the target kinematics the alpha-beta filter, Kalman filter, Unscented Kalman filter, IMM, AIMM, Dual IMM are available [DKO07]. Besides the kinematics the classification and identification estimation is included. These algorithms base on Bayes theory, Dempster Shafer, Fuzzy sets, or expert systems.

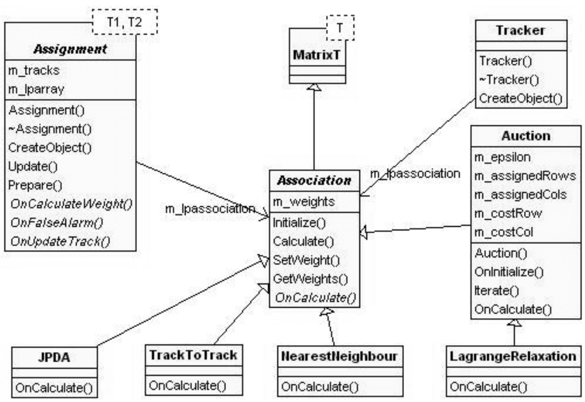

Figure 3: Class model for data association and estimation (subset).

2.4 Data Flow

The processing of the source data shows the benefit of an object oriented approach for a data fusion system. One simply generates for every sensor an instance of "Sensor", which receives and processes the sensor data according to the individual specifications. Only what is need is a multiplexing function, which distributes the data. The sensor tracks are realised through instances of type "SensorTrack" while the sensor track containers corresponds to instances of "SensorTrackArray". Assuming that a sensor delivers only raw plots the integration of a sensor tracker is also possible. Then the instances of "SensorTrack" are produced by a sensor tracker according to the sensor specific attributes. To implement the multisensor tracking each new instance of "Sensor Track" calls a new instance named "FusionTrack" or updates an existing one.

This instance is responsible for the processing of those sensor tracks, which it has subscribed before. Therefore, it is possible to use multiple multisensor trackers in the system which works in parallel. A multisensor tracker contains a container for the fused tracks, called "FusionTrackArray", which refers to all tracks which are candidates for track update. Whenever new sensor data arrives the estimation algorithms are called from the association process while calculating likelihoods matrices (tensors) for the association process. The updated "FusionTrack" calls the kernel and the kernel calls the output interface. Using this multisensor trackers it is easy to design a complex fusion process through a fusion tree, whose nodes are different sensors and multisensor trackers. All sensor and multisensor trackers are configured by a XML file. The XML file configuration determines which alignment, data association and estimation methods are used. The object model is therefore near to the real world environment and allows very flexible adaptations of the fusion kernel (figure 4).

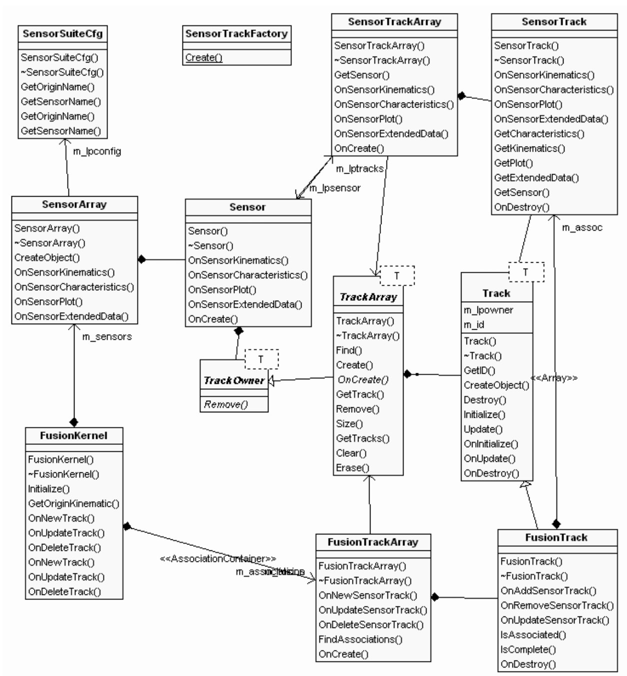

Figure 4: Class model for sensors, sensor trackers and multisensor tracker.

3 System Design, Simulation and Testing

The design of complex data fusion systems is supported by an graphical GUI interface. For illustration purposes figure 5 shows the design of a data fusion systems consisting of two separated platforms. Each platform carries its own sensor and onboard the platform the sensor data are processed with a sensor tracker.

The data of booth platforms are fused with a multisensor tracker, which operates on the associated sensor plots founded by the two sensor trackers on the platforms, which perform the surveillance task. All three trackers, i.e. booth sensor trackers and the multisensor tracker uses an IMM and nearest neighbour data association. Also the parameterisation of the estimation algorithms are supported by special GUIs. The resulting configuration is coded within a XML file which is passed whenever the fusion kernel is started up. However it is translated into binary information, so that the configured data fusion system is a real-time application. The data fusion kernel itself is in ANSI C++ and therefore available for different operating systems. Further features address the topics simulation, test and evaluation of data fusion systems.

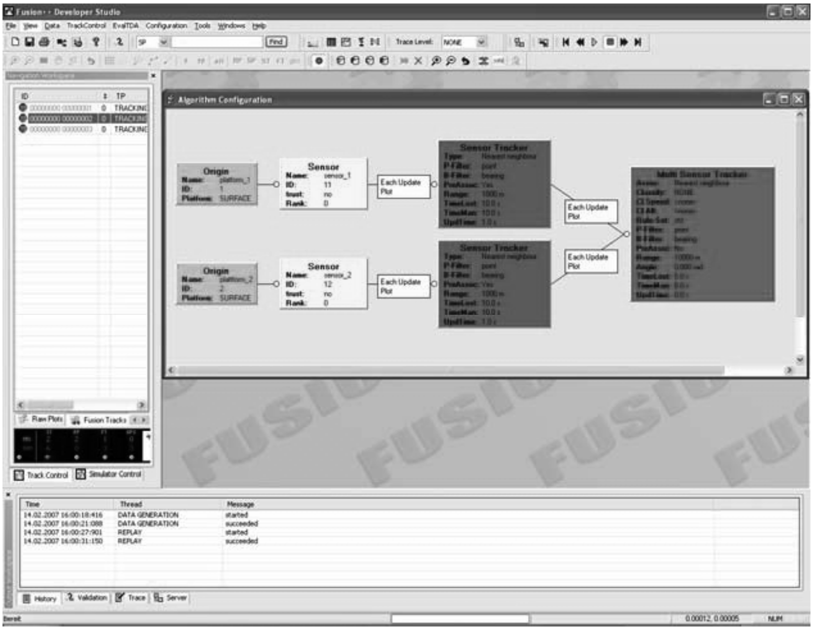

Figure 5: Example of system design GUI.

References

[DKO07] K. Dästner, T. Kausch, F. Opitz: An Object Oriented Development Suite for Data Fusion: Design, Generation, Simulation and Testing. Fusion 2007, Québec, Canada, 2007.

[OHK04] F. Opitz, W. Henrich, T. Kausch: Data Fusion Development Concepts within complex Surveillance Systems, Fusion 2004, Stockholm, Sweden.

[BS01] C. L. Bowman, A. N. Steinberg: A systems engineering approach for implementing data fusion systems. In D. L. Hall, J. Llinas (editors), Handbook of multisensor data fusion, pp. 16-1 – 16-39, CRC Press, Boca Raton, 2001.

[OK05] F. Opitz, G. Kouemou: An Unified Approach to Radar System Simulation & Development, Proc. International Radar Symposium IRS 2005, Berlin, Germany, 06-08 September 2005.